REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
August 2011	Conference Paper- Post Print	October 2008 – September 2010

4. TITLE AND SUBTITLE

QoS-T: QUALITY OF SERVICE THROTTLING TO ELICIT USER COOPERATION IN COMPUTER SYSTEMS

5a. CONTRACT NUMBER
IN-HOUSE

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62702F

6. AUTHOR(S)

Vidyaraman Sankaranarayanan, Shambhu Upadhyaya, Kevin Kwiat

5d. PROJECT NUMBER
23G4

5e. TASK NUMBER
IH

5f. WORK UNIT NUMBER
01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RIGG
525 Brooks Road
Rome NY 13441-4505

University at Buffalo
Dept of Computer Science & Engineering
Buffalo NY 14260

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory
26 Electronic Parkway
Rome NY 13441

10. SPONSOR/MONITOR'S ACRONYM(S)
N/A

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TP-2011-33

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA #: 88ABW-2008-1165. Date Cleared: December 2, 2008.

13. SUPPLEMENTARY NOTES
Proceedings of the International Conference on Mathematical Methods, Models and Architectures for Computer Network Security.

14. ABSTRACT
While there exist strong security concepts and mechanisms, implementation and enforcement of these security measures is a critical concern in the security domain. Normal users, unaware of the implications of their actions, often attempt to bypass or relax the security mechanisms in place, seeking instead increased performance or ease of use. Thus, the human in the loop becomes the weakest link. This shortcoming adds a level of uncertainty unacceptable in highly critical information systems. Merely educating the user to adopt safe security practices is limited in its effectiveness; there is a need to implement a technically sound measure to address the weak human factor across a broad spectrum of systems. In this paper, we present a game theoretic model to elicit user cooperation with the security mechanisms in a system. We argue for a change in the design methodology, where users are persuaded to cooperate with the security mechanisms after suitable feedback. Users are offered incentives in the form of increased Quality of Service (QoS) in terms of application and system level performance increase. User's motives and their actions are modeled in a game theoretic framework using the class of generalized pursuit-evasion differential games.

15. SUBJECT TERMS
Game Theory, Human Factor in Security, Quality of Service, QoS, Computer Security, Threat Model

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			KEVIN A. KWIAT
U	U	U	UU	17	19b. TELEPHONE NUMBER (*Include area code*) N/A

QoS-T: QoS Throttling to Elicit User Cooperation in Computer Systems

Vidyaraman Sankaranarayanan
Microsoft
1 Microsoft Way
Redmond, WA 98052
krsna@acm.org

Shambhu Upadhyaya
Dept. of CSE,
SUNY @ Buffalo
Buffalo, NY 14260
shambhu@cse.buffalo.edu

Kevin Kwiat
Air Force Research Laboratory
525 Brooks Road
Rome, NY 13441
kwiatk@rl.af.mil

Abstract. While there exist strong security concepts and mechanisms, implementation and enforcement of these security measures is a critical concern in the security domain. Normal users, unaware of the implications of their actions, often attempt to bypass or relax the security mechanisms in place, seeking instead increased performance or ease of use. Thus, the human in the loop becomes the weakest link. This shortcoming adds a level of uncertainty unacceptable in highly critical information systems. Merely educating the user to adopt safe security practices is limited in its effectiveness; there is a need to implement a technically sound measure to address the weak human factor across a broad spectrum of systems. In this paper, we present a game theoretic model to elicit user cooperation with the security mechanisms in a system. We argue for a change in the design methodology, where users are persuaded to cooperate with the security mechanisms after suitable feedback. Users are offered incentives in the form of increased Quality of Service (QoS) in terms of application and system level performance increase. User's motives and their actions are modeled in a game theoretic framework using the class of generalized pursuit-evasion differential games.

Keywords: Game theory, Human factor in security, Quality of Service, Computer Security, Threat model

1. Introduction

Traditionally security and quality of service (QoS) have been perceived as only orthogonally achievable goals. The enforcement of security is thought to be a performance obstacle, and guaranteeing QoS is thought to require the relaxation of security mechanisms [4]. These are the misconceptions that drive normal users to bypass or relax the security mechanism in place. Unaware of the implications of their actions, they seek, instead, increased performance or ease of use. Using ineffective passwords [2], disabling critical security features, installing untrusted software [5], and not applying security patches in a timely manner are a few instances of user level lapses that impede security. According to a survey [19] conducted by McAfee, users, in the hope of gaining an immediate functionality, may recklessly download and install shareware programs on their company issued laptops or bring in their own

gadgets to the workplace. Such lapses are unacceptable in highly critical information systems. However this brings out an interesting point. If the human in the loop proves to be the weakest link, regardless of the sophistication and strength of the security measures taken, their implementation and particularly, their enforcement in a system must be of critical concern. We argue that enforcement of these security measures requires reversing or in the least, manipulating the above misconceptions. Recently there has been some work done on viewing security as one aspect of QoS and in turn, seeking a symbiotic relationship between the two system interests.

In this paper, we aim to exploit the obvious interdependence between quality of service and security in order to improve overall system security, *particularly in interactive systems*. Unlike previous approaches that tend to address a single threat vector [4, 6, 10], the work in this paper describes an underlying approach, similar in theme to [9], that may be used in interactive systems. Given that users prefer greater performance and increased quality of service, we propose a model to prevent security breaches and elicit user cooperation with the security mechanism. The focus of this paper is towards dealing with this class of problems where the system security level is degraded due to user action/inaction. We take the view that all failures due to user action or inaction have to be treated as engineering failures, instead of being ignored. We present a game theoretic model intended to directly counterbalance this risk. The purpose of the model is twofold:

- elicit user cooperation with the security mechanisms in place by gracefully providing incentives to end users as they provide demonstrable evidence of cooperation with the security subsystem
- punish potential intruders who refuse to cooperate with the security subsystem with a reduced QoS

This approach is similar to [7], where hostile users in a wireless ad-hoc network are punished by active jamming. Our approach enjoys two main benefits: It encourages legitimate users to cooperate with security mechanisms as well as deters rogue users by proportionally degrading QoS in light of suspected security breaches.

The underlying concept of degrading performance in case of observed security problems is present in different forms. In the area of network-security, for example, a server may gradually start dropping connections or reducing the QoS to stop a DoS attack or delay the propagation of Worms. We extend this idea to service throttling in order to address the weak human factor. Our mechanism is applied in cases where there is no *absolute certainty* that there is an attack (malicious traffic in the case of a DoS and improper user activity in our case). Degrading performance is done for two reasons: delaying the attack (if there is one in progress) and ensuring user level compliance to the security policies.

The rest of the paper is organized as follows. Section 2 discusses related work on addressing the weak human factor. Section 3 presents the QoS degradation model and the flow of control in the model. Section 4 presents a proof-of-concept simulation that illustrates the usage of this model. Concluding remarks are given in Section 5. The appendices contain the differential games used in the underlying QoS degradation model of Section 3.

2. Related Work

Researchers in [2], [9] and [22] all arrive at the general conclusion that users may be careless and unmotivated when it comes to system security; however they argue that the fault lies ultimately with the design and implementation of these security mechanisms. Adam and Sasse [2] discuss "Users' Perceptions on Security" and the importance of accounting for these perceptions. Dourish et al. [9] argue the importance of creating degrees of security as opposed to the traditional "all-or-nothing" black-box approach. In this way, users naturally distinguish between highly sensitive versus less-sensitive information systems and this manifests itself through different behavior in these different environments. Adam [2] and Sasse [22] also emphasize the importance of *removing* the transparency from security tools, particularly in highly critical systems, and *actively involving users* in the security cycle. With these criteria in mind, in this paper, we developed a graded QoS model to make users personally accountable for the state of the system. Linn [16] introduces a parameter intended to manage the level of protection provided by a security mechanism. Irvine et al. [14, 15] define security as a constructive dimension of QoS rather than an obstacle. Our approach translates variable security levels directly into variable QoS levels returned to the user. In this way, there is a tangible motivation for the user not to circumvent the security mechanism.

The problem of the weak human factor has been researched in the same vein, by using fear appeals [29] or by forcing the user to interrupt their workflow [31] for the 'greater good.' Generic approaches have also been proposed by means of equating safety properties to security properties [3]. Certain online banking systems ask in addition to the password, personal information about the user (like SSN number, Drivers license, etc.) during a login procedure. However such measures are geared only towards malicious users and do not involve legitimate users in the security subsystem. A model called 'safe staging' [30] by Whitten and Tygar extends this notion to legitimate users, where a system restricts the rights of Java applets (the service quality) in response to users' demonstrated understanding of the security implications. As users become more familiar with the security issues, the service quality is increased. Our model extends this notion a step further by incorporating a monitoring and feedback control mechanism to involve legitimate users in a constructive manner. This work represents a foundational shift in the very approach to addressing the weak human factor [21], by addressing user actions and treating them in the same manner as user input would be treated for buffer overflow attacks.

3. QoS Throttling (QoS-T) Model

Essentially, the problem we seek to solve is an important one, but has eluded a technical solution due to a variety of reasons. Primary among them is the *act of interference* and *lack of control*; any technological solution that seeks to remedy the weak human factor does so by means of either interfering in the workflow of the user or taking away control of the system from the user, or a combination of both factors. These two factors irk users; security designers have not found the correct balance or

an alternative. Our approach is a combination of these two factors, but in a very *gradual and subtle manner* with appropriate feedback, thereby giving the user complete control at every stage, with minimum to zero interference to the workflow. The nature of this problem involves understanding and quantifying user actions, their incentives and ensuring an optimal state where the user objectives are met and the system security is also maintained. Thus, the problem may be viewed as one of balancing the objectives of the user and that of the system. In such a situation, game theoretic models apply naturally.

3.1 Why Game theory?

We have chosen the class of generalized pursuit-evasion differential games for modeling this problem. Game theory helps model a set of 'selfish' and 'rational' players who act in a setting solely for their own advantage. Users in our setting can be said to act selfishly to improve their own QoS. Game theoretic models have been used to infer the incentives of attackers [20] based on their perceived incentives. In this work, we use a game theoretic model to provide incentives to the user in order to elicit cooperation. The purpose of the game theoretic model is to derive a measurable quantity out of the user's actions that can be given as a feedback to the security mechanism. User's actions in the game theoretic setting are equivalent to strategies of a player. The security mechanism can use the payoff function of the game to adjust the QoS. The advantage of modeling the user/resource/security-mechanism scenario as a differential game is that it allows for a flexible definition of the act of "a user accesses a resource." While this definition is abstract at the level at which this model is described, it can be properly interpreted and applied on the specific security domain where its application is relevant. The model is self enforcing; we do not make any assumptions about the coordination between the players of the game. Users in a system need not be aware of the model, nor are they required to consciously participate in any 'game.' Differential game-theory also has the notion of 'continuous' play, which makes it conducive to use it in situations as these. Lastly, the usage of game theoretic notions allows us to specify notions of strategy or best responses of the participating players (the users and the system) thereby leading to good mechanism design that elicits cooperation from users as a natural process. The reader is referred to [8, 17] for a more detailed exposition on game theory. The specifics of the games used in our model are described in the appendix.

3.2 Types of Users

The threat posed by legitimate users in an organization has appropriately been labeled as "The Enemy Within" [19] in a recent survey by McAfee Corporation (http://www.mcafee.com). We can divide the user broadly into two different categories:

- Type I: A Legitimate User – This category of users includes legitimate and authorized users of the system. These users log into the system and execute workflow processes according to their roles. According to the McAfee Survey [19], such users are varyingly labeled as "The Security Softie", "The Gadget

Geek" or "The Squatter." While they do not have any stated intentions to disrupt the system, their actions nonetheless endanger the system. For example, these users do not have any idea of the threat model of the system and hence, may not implement the best practices suggested by the organization.

- Type II: A Legitimate, but Malicious User – Similar to Type I users, users in this category are legitimate, i.e., they possess authorized credentials to log into the system. However, their goal is to disrupt the system, either through a self inflicted cataclysmic system compromise or through slow poisoning attacks like leaking confidential information about the organization to its competitors. According to the Survey [19], such users are labeled as "The Saboteur."

Let us first examine the challenges that researchers and designers face when dealing with the weak human factor. In any system, users perform actions towards fulfilling their roles. The notion of actions is an abstract one that can be generalized to most, if not all, systems. Actions can be split in the following manner.

- *Action Type I* – the fundamental user actions required for the workflow: These fundamental actions are defined by the user's role in the environment. For example, a graphics designer will need to use some photo/video editing software. In addition, a device like a tablet may need to be connected to the computer via the USB interface for rendering hand sketches.
- *Action Type II* – Ancillary actions required for the fundamental actions to work: For example, exploring the hard drive is a prerequisite for most job roles. In addition, connecting USB devices, burning images onto a CD may be in this list for a graphics designer.
- *Action Type III* – These are actions that are not predefined like Action types I and II. These actions are the ones that users normally execute without any restrictions, since they do not fall under the purview of 'restricted objects.' They might have the potential to disrupt the working of the system, or may be inimical to the individual. Examples of such actions include clicking on a potential phishing link in an un-trusted/unsigned email.

For those actions that are relevant to the security of the system, there exists an easy or an efficient manner of performing them. For example, choosing a password is an (one time) action that users have to perform when registering into the system. The easy way is to choose a password that is easy to remember (and hence easy to guess/crack). The efficient way, on the other hand, is to choose a complex password that is tough to remember. Similar is the situation with security updates; it is easy to ignore them while it is efficient to update the system. For reasons that are mostly context and domain specific, users prefer to perform only the easy action, and not the efficient one. Viewing the interaction between the user and the system as a set of easy vs. efficient actions, where the easy action is most often the inefficient one, provides us a global view to look into this issue. Thus the main challenge for human centered security schemes is to ensure that users perform the efficient action with awareness of the consequences of their actions. Viewing these actions under the three prisms provides us one methodology to address the human factor related security issues.

3.3 Process Flow

The flow of control for the QoS throttling model is shown in Figure 1.

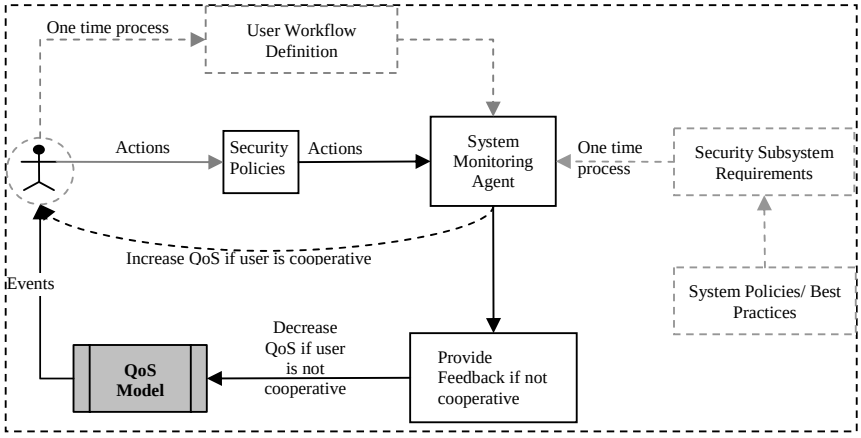


Fig. 1. Control Flow

We first start with the security mechanism and derive its requirements. Towards this, we may use the systems' best practices as a guide. Concurrently, we define the user's workflow process. These two steps are one-time processes which ensure that (a) at no cost is the users workflow adversely affected and (b) the Monitoring agent is aware of the security subsystem's expectations. The user's session then proceeds as usual, where every user action is first filtered by the security policies in the system. These filtered actions are monitored by the monitoring agent, which decides if the actions are in conformance with the security subsystems requirements. If the user is cooperative, he is rewarded with a gradual increase in the QoS. If the user is not cooperative, a feedback is given (similar to 'Install Updates' dialog box in the windows environment, etc.) with a request to cooperate. If the user still blatantly refuses to cooperate, the gradual application and context specific QoS throttling is initiated.

3.4 QoS-T Model 1: Exponential Back-off

Given a singleton process, we discuss a simplistic exponential back-off model to evaluate a decreasing time delay. This time delay could be used as a parameter to the artificial sleep statements (or any other context specific delay). This model is useful in situations where the system (and its threat model) is simple enough with an automated mechanism that classifies the user. We define the QoS throttle through a simple equation:

$$f(x) = (1 - x)\cdot e^{1/x} \qquad : x \in [0,1] \tag{1}$$

where *x* is the quantitative input that grades the users classification and *f(x)* the time delay (in some appropriate time units) that is imposed by the system. The value *x* can be the trust level of the user in the system, a real number between 0 and 1, where 0 represents a untrustworthy user and 1 a trustworthy user. For those systems that have a mechanism to detect their security level (or trust level of users), the exponential back-off model may be used. For example, the *Compensatory Trust Model* [28] is an automated trust evaluation mechanism specifically designed for users in an authenticated system. The exponential back-off model has the advantage of simplicity, clear intuition and an easy translation to an implementation. Also, the intuition behind it may be changed depending on the system (and the users) to derive other ancillary models (different distributions) that may perform better for particular systems.

3.5 QoS-T Model 2: Game Theoretic Approach

Given a workflow process with multiple sub-processes, we present a game theoretic model that can be used to gradually reduce the QoS of a sub-process and tag the user as proceeding gradually from a non-cooperative user to a malicious user during the workflow. We leverage on two well-studied problems of game theory – the two player differential game of "Guarding a Territory" [12] and the "Dolichobrachistochrone" [11].

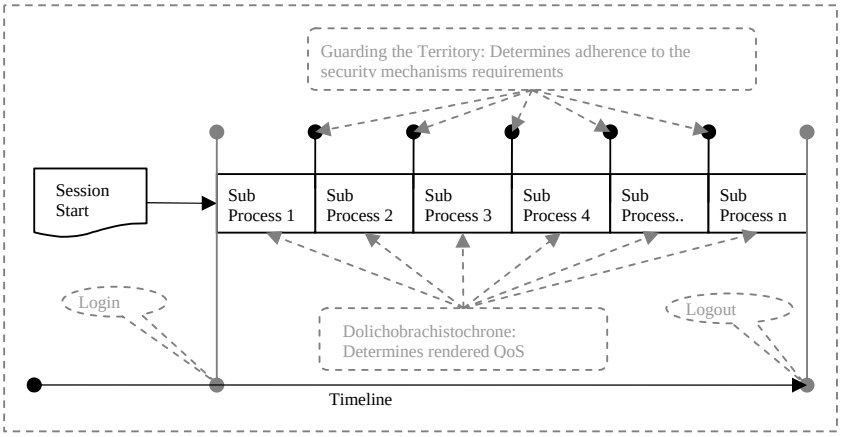


Fig. 2. Modeling the Workflow and constituent Sub-Processes

The entire timeline of the users' actions in a session is split into many fragments. Each fragment represents some sub-process in the workflow (e.g., the execution of a process/application). The process of changing the QoS is equivalent to varying the rate/difficulty with which the user can access the resource or complete execution of the process. The Dolichobrachistochrone game models each fragments of the timeline, i.e., the resource-access event. After every resource access, we need to determine the security state of the system (or in other words, classify the user and

hence infer the security state of the system). The second differential game of "Guarding the territory" models the security state of the system. This game also models the point where the *noncompliant user becomes a malicious user*. These two games are chained to provide proper feedback so that the output of the Dolichobrachistochrone game is the input to "Guarding the territory." This process is shown in Figure 2. The Dolichobrachistochrone game is also called a supergame, within which repetitions of the smaller 'guarding the territory' game is played. The game-theoretic model is more involved, with a greater emphasis on system specifics and an inbuilt mechanism for user classification. A detailed description of these games is given in the appendix.

The control variables of the two games are chosen depending on the system under consideration and the security mechanism. The final step in the model is to chain the two games so that the output of the Dolichobrachistochrone is the input of the "Guarding the territory" game. The payoff of the Dolichobrachistochrone is *x(T)*, which is the distance traveled by the particle P. The player **u** in the "Guarding the territory" game can now travel a distance *x(T)* towards the region **Ω** by a predetermined angle. If the player were to reach the region before the session is over, the security state of the system can be changed and appropriate action can be initiated.

4. Proof-of-Concept Illustration

In this section, we introduce a simplified, yet generalized scenario encountered by network administrators in most IT organizations. We then derive the threat model from a basic social engineering attack and present the application of the game theoretic QoS-T model to this problem, illustrating its practical utility.

4.1 Threat Scenario

An experiment [26] was conducted by "The Training Camp" where commuters in London were offered free CD's with special Valentine Day's promotion. Despite a clear warning, most employees apparently inserted the CD and ran the program (which displayed a warning against such actions). In a similar vein, another experiment [25] (akin to a social engineering penetration testing) revealed that free USB disks which were 'discovered' by employees were blindly inserted into computers, thereby triggering the execution of a (potentially malicious) program.

The situation is similar with the case of downloading and installing programs from the Internet. For example, consider the process of downloading and executing a file from the Internet. The user launches a browser, connects to the web site that hosts the file (or is redirected to the site), downloads the executable and then executes it. Assume that the systems' best practices state that unless a downloaded executable is signed by a trusted publisher, it is preferable to not execute it. This typical sequence of operations initiated by the user can be broken down into sub-processes, each of which plays a role in the complete operation. This example brings out the following points:

(a) The process of downloading and running untrusted executables is a manifestation of the weak human factor.
(b) This process is not part of the user's workflow in the organization.
(c) The entire process can be split into a number of sub-processes:
    a. Browsing to an untrusted zone
    b. Initiating a File download
    c. Executing the file

For the sake of illustration, we assume that executing untrusted executables in the current user context is not completely prohibited, but is undesirable.

With this scenario, let us explore how the new paradigm can be applied. We have two levels here. First we want to throttle the service quality, but not affect the user's legitimate workflow. Secondly, we would like to use the additional CPU cycles gained to perform some useful work, in terms of increasing system performance and security. To degrade the application level QoS, the browser could be slowed down in a number of ways (inserting artificial sleep statements in the browser process, slowing down the network bandwidth available to the browser process, etc.). This degradation is initiated only after a proper feedback is provided to the user, warning him to refrain from the actions. This degradation by no means affects the system performance (if there are any background processes running) and provides the developer an opportunity to insert (for example) security logging statements, like logging the site where the browser navigated to, the plug-ins activated by the site, etc. After the application has been downloaded, the user could be given an option to run the application inside a sandbox with restricted permissions, or run the application with less than normal privileges. Additionally, the application level QoS could be degraded by inserting artificial sleep statements in an approach similar to [30].

4.2 Threat Model: Multiple Untrusted Applications Execution

In this threat model, we envisage a scenario where users are required to specify in their workflow patterns the most commonly used applications in a typical session. This represents a secure and controlled environment such as the military operations or a secure and compartmentalized job in an industry. As mentioned in Figure 1, specifying the workflow is a onetime process. If the applications users execute fall within the purview of the workflow, they are accorded a high application level QoS. As they execute applications outside the workflow specification (possibly due to malicious intent or due to an impersonation attack), the QoS is gradually reduced. When the number of applications outside the workflow specification exceeds a limit defined by the users trust level, the user is declared malicious. This clearly illustrates the game theoretic model; one game is used to determine the QoS and the other is used to determine when the non-cooperative user becomes a malicious user.

The QoS degradation for this threat model is similar to the concept of penalizing specific system processes, which is the approach by Somayaji and Forrest [24], where an exponentially increasing delay (artificial sleep statements) was introduced between system calls.

The final step is translating the model to the actual timing details. We fixed the distance of the user from the territory $\boldsymbol{\Omega}$ (Figure 6 in the appendix) to be the maximum number of unauthorized applications for a standard user (x_o = 7 units in Eq.

5 in the appendix). As we shall see, users are tagged malicious if their trust level is low or never tagged as malicious if their trust level is high, even if they exceed the maximum value set for the standard user. The users' trustworthiness ($\mathbf{U_T}$) was varied between 0 and 1 ($0 < \mathbf{U_T} \leq 1$). ω in Eq. 5 in the appendix was set to 1. Finally the delay time (T) in the Dolichobrachistochrone game is inversely proportional to $\mathbf{U_T}$ ($T = \alpha / \mathbf{U_T}$). One time unit is set to 10 milliseconds for this plug-in. These assignments finally reduced the model to evaluation of the Dolichobrachistochrone game Eq. 5 in the appendix, which now reads as follows:

$$x(T) = x_o - \frac{T}{2} + T - \frac{1}{2}\sin(T) \qquad (2)$$

The variable α (in $T = \alpha / \mathbf{U_T}$) for each action was set and subsequently increased according to Table 1. Subsequent values of x_o were assigned to the previous values of $x(T)$ as calculated by Eq. 5 in the appendix. We varied the users' trust level and plotted the time delay as well as the number of unauthorized applications it would take for the user to execute to penetrate the territory. The resulting action by the security subsystem depends on the domain. The plug-in raised an administrative alert when the territory was reached by the user.

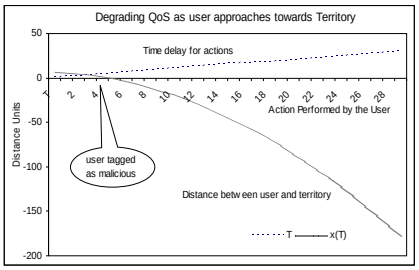
Table 1. Values of U_T and corresponding α

Trust Level (U_T)	**Alpha** (α in $T = \alpha / U_T$)	**Figure**
0.1	Initially set to 0.1 and increased by 0.1	3.a
0.3	Initially set to 0.1 and increased by 0.1	3.b
0.7	Initially set to 0.15 and increased by 0.15	4.a
1.0	Initially set to 0.2 and increased by 0.2	4.b

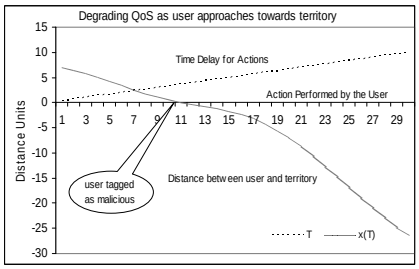
Figure 3 shows the time delay for low values of user trust levels and the number of actions (or equivalently, the number of untrusted applications executed) it takes for the users to transit from a non-cooperative user to a malicious user.

QoS Degradation for $U_T = 0.1$: Figure 3.a shows the time delay rate (T) and the progress of the user towards the territory ($x(T)$) for a trust level of 0.1. Since the users trust level is very low, the user rapidly progresses towards the territory, indicative of his low trust level; he is tagged as malicious (at the point where $x(T)$ crosses y = 0) by the fifth unauthorized application.

QoS Degradation for $U_T = 0.3$: Contrast this with Figure 3.b, which shows the same plot for a trust level of 0.3. The time delay rate is still the same (α is the same), but the user approaches the territory slowly, indicative of an increased trust level, and is tagged as malicious only by the 11th unauthorized application, as opposed to the fifth one in Figure 3.a. Figure 4 shows the time delay for high values of user trust levels. In this case, we note that the user does not actually penetrate the territory, indicative of the high trust level. Instead, there is a gradual oscillatory movement due to the sinusoidal component in Eq. 5 in the appendix.

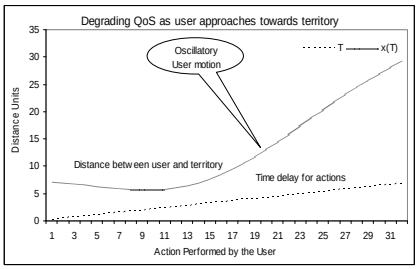


(a) QoS degradation for $U_T = 0.1$

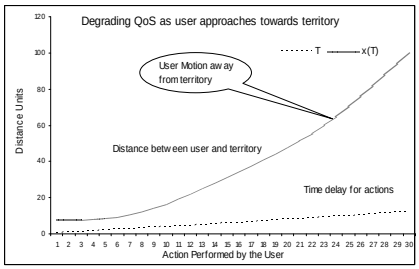


(b) QoS degradation for $U_T = 0.3$

Fig. 3. QoS degradation for low values of user trust level



(a) QoS degradation for $U_T = 0.7$



(b) QoS degradation for $U_T = 1$

Fig. 4. QoS degradation for high values of user trust level

QoS Degradation for $U_T = 0.7$: As illustrated in Figure 4.a, the user initially approaches the territory since the session scope is not complied with. Due to the sinusoidal component in Eq. 5 in the appendix, the initial approach towards the territory is replaced with a movement away from the territory. We interpret this sinusoidal oscillation as follows: This user is not deemed malicious at any point of time, due to his high trust level. But, we also set *a lower time delay rate* as the system expects him to be cooperative and security conscious due to his high trust level. For example, the time delay for this user at the end of the 11th unauthorized application action is 2.35 time units (23.5 milliseconds) while the time delay for the user in Figure 4.b ($U_T = 0.3$) is 3.66 time units (36.6 milliseconds).

QoS Degradation for $U_T = 1$: Here we illustrate a situation where the user is completely trustworthy. In Figure 6.b, the user does not even approach the territory (indeed, he moves away from it) since he is completely trustworthy. However, the *time delay is made higher* (0.2 units). The time delay at the end of the 11th action for this user is 4.4 units (44 milliseconds). Such progressively high time delays virtually render the unauthorized applications inoperable (for the user). Note that the act of setting higher time delays for highly trusted users is intuitive and logical, since trusted users in mission critical areas are expected to be aware of the security subsystem and hence, cooperative. Their high trust levels ensure that they are not tagged as malicious at any point in time, *a privilege they earn at the cost of actively cooperating with the security subsystem*. A simple scheme to ensure that users consistently approach the

territory as time progresses is to provide a feedback loop and lower their trust level progressively. This aspect, however, is out of scope of this paper, for trust assignment and management is another research area by itself.

5. Conclusion and Future Work

Farmer's Law states, "The security of a computer system degrades in direct proportion to the amount of use the system receives." Farmer self-proclaimed this as his law in a survey on the security of Key Internet Hosts, highlighting the fact that users are often the greatest risk to system security. In a similar vein, Schneier [23] states that the very interaction between humans and systems forms the greatest risk to IT systems. For specific threat models (like weak passwords, phishing, social engineering, etc.), there are specific solutions. But the greater problem of involving the users in the security loop has remained unaddressed so far. The reason for this lies in the misconception that QoS and security are orthogonally achievable goals. In highly-critical information systems the need to appeal to these users and elicit their cooperation is paramount. Trading application level QoS in terms of transparency/ease of usage for user involvement with the security mechanisms in place is justified and in fact, necessary. The solution advocated in this paper is a graceful degradation of the rendered application specific QoS that the user perceives *in the face of a conspicuous lack of cooperation*. For example, consider the case of data breaches in corporate environments; a recent article in the Wall Street Journal states [32] states that data breaches are on the rise; most often, the data breaches are not detected immediately and offenders are rarely, if ever, held accountable. The QoS-T framework proposed in this paper could be viewed as a contractual requirement by the customer of businesses; it may be viewed as a mechanism to correct complacency by corporate members' in-situ. Any complacency by businesses (and their employees) in applying appropriate security measures towards data protection would lead to a lowering of QoS, which in turn, would directly affect productivity (and hence, would affect the "sacred" bottom-line). Thus conformance to security measures will not be limited to merely a moral code but enforced with a monetary means. The application of game theoretic models to practical scenarios will lead us into interesting problems [18] which have to be resolved in a context specific manner. Although it is debatable if such a model will *really* ensure user cooperation, we hope that in the same manner a user types his password carefully the second time to avoid typographical mistakes (and hence additional delays in password systems), the implementation of this model will encourage the user to cooperate and actively participate with the security subsystem.

References

[1] *DoD Directive 8500.1 "Information Assurance (IA)"*, 2002.
[2] Anne and S. Martina Angela, *Users are not the enemy*, Commun. ACM, 42, pp. 40-46, (1999)

[3] S. Brostoff and M. A. Sasse, *Safe and sound: a safety-critical approach to security, Proceedings of the workshop on New security paradigms*, ACM Press, Cloudcroft, New Mexico (2001)
[4] S. Brostoff. and M. A. Sasse., *Ten strikes and you're out: Increasing the number of login attempts can improve password usability, CHI 2003 Workshop on Human-Computer Interaction and Security Systems*, Ft. Lauderdale, FL, USA (2003)
[5] CERT, *CERT® Advisory CA-2000-04 Love Letter Worm: http://www.cert.org/advisories/CA-2000-04.html* (2005)
[6] H. Cheryl and E. Chinedu, *Increasing security and usability of computer systems with graphical passwords, Proceedings of the 45th annual southeast regional conference*, ACM Press, Winston-Salem, North Carolina (2007)
[7] Dave Levin, *Punishment in Selfish Wireless Networks: A Game Theoretic Analysis*, P*roceedings of Economics of Networked Systems, NetECON* Ann Arbour, Michigan (2006)
[8] M. Davis, *Game Theory: A nontechnical introduction*, Dover (1983)
[9] P. Dourish., R. E. Grinter., B. Dalal., J. D. Flor. and M. Joseph., *Security Day-to-Day: User Strategies for Managing Security as an Everyday, Practical Problem*, Institute for Software Research, University of California, Irvine (2003)
[10] B. Francesco, G. Daniele and P. Claudia, *User authentication through keystroke dynamics*, ACM Trans. Inf. Syst. Secur. Vol. 5., pp. 367-397 (2002)
[11] Freedman, *The Dolichobrachistochrone Game*, *Differential Games*, John Wiley & Sons, Inc., pp. 107, (1971)
[12] Freedman, *Guarding a Territory*, *Differential Games*, John Wiley & Sons, Inc., pp. 29, (1971)
[13] M. Howard, *Browsing the Web and Reading E-mail Safely as an Administrator*, *MSDN* (2004)
[14] C. Irvine, T. Levin and E. Sypropoulou, *Security as a Dimension of Quality of Service in Active Service Environments*, *International Workshop on Active Middleware Services*, San Francisco, CA (2001)
[15] C. Irvine and L. Timothy, *Quality of security service*, *Proceedings of the 2000 workshop on New security paradigms*, ACM Press, Ballycotton, County Cork, Ireland (2000)
[16] J. Linn, *Generic Security Service Application Program Interface, IETF Request for Comments* (1993)
[17] R. D. Luce and H. Raiffa, *Games and Decisions*, Dover (1989)
[18] R. Mahajan, M. Rodrig, D. Wetherall and J. Zahorjan, *Experiences applying game theory to system design*, *Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, ACM Press, Portland, Oregon, USA (2004)
[19] McAfeeCorporation, *The Enemy Within; http://www.theregister.co.uk/2005/12/15/mcafee_internal_security_survey/*, (2005)
[20] Peng Liu, Wanyu Zang and Meng Yu, *Incentive-based modeling and inference of attacker intent, objectives, and strategies*, ACM Trans. Inf. Syst. Secur., Vol 8, pp. 78-118, (2005)

[21] S.Vidyaraman, M.Chandresekaran and S.Upadhyaya, *Position: The User is the Enemy*, *Proceedings of the New Security Paradigms Workshop*, New Hampshire, USA (2007)
[22] M. A. Sasse, *Computer Security: Anatomy of a Usability Disaster, and a Plan for Recovery*, *CHI 2003 Workshop on Human-Computer Interaction and Security Systems*, Ft. Lauderdale, FL, USA (2003)
[23] B. Schneier, *Secrets and Lies: Digital Security in a Networked World*, John Wiley & Sons, Inc., New York (2000)
[24] Somayaji and S. Forrest, *Automated Response Using System-Call Delays*, *Usenix Security Symposium* (2000)
[25] S. Stasiukonis, *Social Engineering, the USB Way: http://www.darkreading.com/document.asp?doc_id=95556&WT.svl=column1_1* , *Dark Reading*, Secure Network Technologies Inc (2006)
[26] W. Sturgeon, *Proof: Employees don't care about security: http://software.silicon.com/security/0,39024655,39156503,00.htm*, *Silicon.com* (2006)
[27] R. Tzur, *SandboxIE, http://www.sandboxie.com/* (2006)
[28] S. Vidyaraman and S. Upadhyaya, *A Trust Assignment Model based on Alternate Actions Payoff*, *Proceedings of the Fourth International Conference on Trust Management (iTrust '06)*, pp. 339-353, *Volume 3986*, Pisa, Italy (2006)
[29] D. Weirich and M. A. Sasse, *Pretty good persuasion: a first step towards effective password security in the real world*, *Proceedings of New Security Paradigms Workshop*, ACM Press, Cloudcroft, New Mexico (2001)
[30] Whitten and J. D. Tygar, *Safe staging for computer security*, *HCI and Security Systems Workshop, CHI*, Ft. Lauderdale, Florida (2003)
[31] H. Xia and J. C. Brustoloni, *Hardening Web browsers against man-in-the-middle and eavesdropping attacks*, *Proceedings of the 14th international conference on World Wide Web*, ACM Press, Chiba, Japan (2005)
[32] Wall Street Journal, *Data Breaches Surpass 2007 Level, But Businesses Rarely Are Penalized* (2008)

Appendix

Dolichobrachistochrone: The Dolichobrachistochrone game is a two player differential game where a point mass P in a uniform gravitational field is constrained to move without friction along a given curve γ. This is illustrated in Figure 5. For equation convenience, the gravitational field is in the direction of the positive y axis. The objective of P is to choose a curve so that it reaches the line x = 0 (the y-axis) in minimum time. The other player E has an objective of trying to slow P as much as possible. E has a force ψ that can be applied to slow P from reaching x = 0. The conditions of the game dictate that the particle P will definitely reach the y axis in a finite time. P's objective is to minimize its arrival time to x = 0. E's objective is to maximize the time for P to reach x = 0. In our model, P represents the user and E the security mechanism. For every access to a resource, the user attempts to minimize his time of access. This translates to P minimizing its arrival time to x = 0. The security

mechanism (E in the game) attempts to vary the rendered QoS according to the force ψ. Figure 5.a shows the particle P falling through the curve γ towards the y axis (x = 0). Figure 5.b shows the player E with an opposing force ψ. The equations of motion for the particle P are described in [11]. The payoff of the game is the distance traveled by the particle P.

$$P(\psi) = x(T) \quad \textbf{(3)}$$

where T is the time for P to reach x = 0 and ψ is a positive constant (ψ = EB = EC in Figure 5.b).

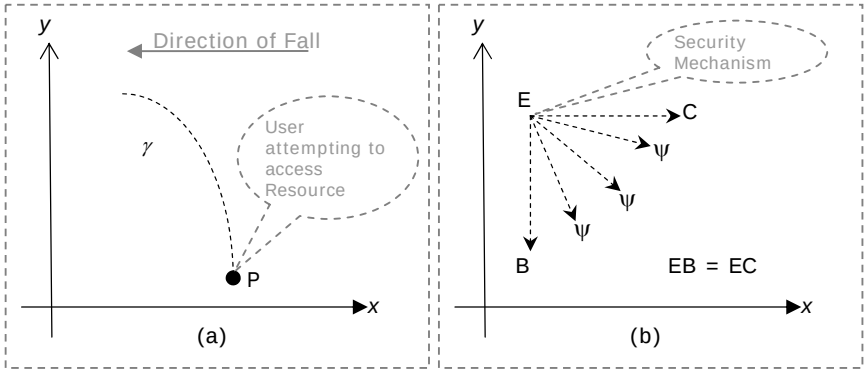


Fig. 5. Dolichobrachistochrone Game

The optimal trajectory for the particle P is given by:

$$y(t) = \frac{y(T)}{2}\left(1 + \cos\frac{(T-t)}{\sqrt{y(T)}}\right) \quad \textbf{(4)}$$

$$x(t) = x_o - \frac{\sqrt{y(T)}}{2}t + \omega t + \frac{y(T)}{2}\sin\frac{T-t}{\sqrt{y(T)}} - \frac{y(T)}{2}\sin\frac{T}{\sqrt{y(T)}};\ y_o = y(0) \quad \textbf{(5)}$$

where -1 ≤ ω ≤ 1. The reader is referred to [11] for more details. The Value of the game (which is the payoff under optimal conditions) is *x(T)* that is evaluated from Eq. 5. Hence the security mechanism can choose *T* (which is the time (delay) taken by the user to access the resource) or equivalently, the force ψ based on the system parameters like the value of the resource being pursued ($\mathbf{R_v}$), the trustworthiness of the user ($\mathbf{U_T}$), etc.

Guarding the Territory: This game represents a model in which a player **v** is guarding a territory **Ω** against an invasion by the player **u**, as shown in Figure 6. The motion of **u** and **v** are described by differential equations [12]. The initial conditions are set as **x**(0) = A and **y**(0) = B. As illustrated in Figure 4, player **v**, the Security mechanism, is located at B, while player **u**, the user, is located at A. In Figure 6, the players are initially separated by a distance AB. C is the mid-point of segment AB.

CY^* is perpendicular to AB, with Y^* being the nearest point to the region **Ω** such that Y^*Z^* is perpendicular to Y^*C. Z^* is the point on the region **Ω** that is nearest to the line segment CY*. We denote the distance of any point **x** on the plane to the territory **Ω** as $d(\mathbf{x},\mathbf{\Omega})$. Each cooperative action by the user symbolically takes him farther away from the region **Ω**. The model expects the security mechanism to provide a feedback on the nature of the user's action and a quantitative measure of the same (which is obtained, in this case, from the Dolichobrachistochrone game).

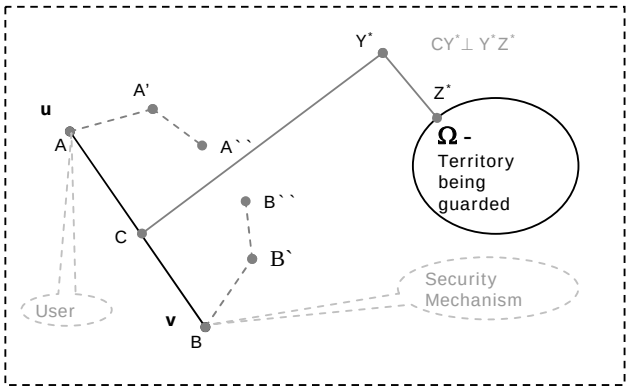


Fig. 6. Guarding the territory

This measure effectively takes the user towards the region **Ω** or away from it. The region **Ω** is the "intrusive" region which the security mechanism can be thought of as trying to protect. The payoff function of the differential game is given as:

$$P(\mathbf{u},\mathbf{v}) = \begin{cases} d(x(\tau), \Omega), & \text{if } \tau < \mathbf{T} \\ N, & \text{if } \tau = \mathbf{T} \text{ and } x(\tau) \text{ lies on the same side of } CY^* \text{ as A} \\ 0, & \text{if } \tau = \mathbf{T} \text{ and } x(\tau) \text{ lies on the same side of } CY^* \text{ as B} \end{cases} \quad (\mathbf{6})$$

where $N > d(Y^*, \mathbf{\Omega})$. In a typical equilibrium strategy, every motion of **u** towards the region **Ω** is matched by **v** by a similar mirror image move across CY^* as indicated by the dotted lines in Figure 4. For instance, the move AA` is matched by BB`, A`A`` by B`B``. The objective of the player **u** is to minimize the payoff $P(\mathbf{u},\mathbf{v})$ in Eq. 5, whereas player **v** tries to maximize it. Hence, in the original game, if player **u** chose not to come near the territory **Ω**, he is penalized by a payoff N. If **v** did not guard the territory "very well", he is penalized by a payoff of 0.